REMARKS

Claims 1 to 46 are pending after this amendment.

Compliance with 35 U.S.C. §112

The claims, as amended, are submitted to satisfy the requirements of 35 U.S.C. §112.

Compliance with 35 U.S.C. §102

The Office Action alleges that claims 1-11 and 19-46 are anticipated by Pettey (US 7046668). The Applicant respectfully submits that this is incorrect. Pettey, as understood, fails to disclose features of the claims, as discussed below.

Pettey

Importantly, Pettey is primarily concerned with I/O within nodes rather than communication between nodes. In Pettey, exchange of data between different nodes takes place via conventional network communication (e.g. Ethernet (128) or Fiber Channel (130) or 'other' networks).

Pettey describes prior arrangements in which a number of servers each have dedicated I/O controllers (of which Fig. 1 is an example) and also arrangements which include a shared I/O switch so that two or more servers can share I/O controllers (of which Fig. 4 is an example).

Fig. 1 of Pettey, as understood, shows an arrangement where each server (102, 104, 106) has a root complex (108) and a dedicated set of I/O controllers (110, 112, 114). The I/O controllers are connected to corresponding networks (128, 130, 132) by way of switches (122, 124, 126). The communication channel between each root complex (108) and the corresponding I/O controllers (110, 112, 114) is part of the load/store fabric of the root complex (see col. 8, ln. 37-39). Each I/O controller communicates with the corresponding switch using the corresponding protocol (e.g. Ethernet) (see col. 8, ln. 34-35).

In Pettey's systems which include shared I/O switches (see Fig. 4, for example), data is supplied from root complex (412) to a shared switch (420) using a load/store fabric (such as

PCI Express). Figure 4 of Pettey, as understood, discloses a computer system having a number of servers (404, 406, 408). Each server includes one or more processors (410) and a root complex (412). The root complex mediates access of a processor (410) to I/O controllers (440, 442, 444). A shared I/O switch (420) is provided between the root complexes (412) and the I/O controllers (440, 442, 444). The host CPUs and their operating systems communicate with the I/O controllers by way of the shared I/O switch. The I/O controllers have separate sets of control registers for each server (OS Domain) (see col. 16, ln. 32-43).

Packets traveling between shared I/O switch (420) and I/O controllers (440, 442, 444) are modified to include an OS Domain Header (e.g. 1100) (see col. 14, ln. 39-46). The OS domain Header serves to associate a packet with its originating or destination root complex. The fabric (address space) of each server is extended to the I/O controller. The OS Domain header is inserted in between the PCI Express sequence number and header components (see Figure 10). Pettey refers to the modified PCI Express protocol as PCI Express+.

Pettey - Handling of Inter-Node Communication

Consider a packet originating at a root complex (412) of a Pettey server (404, 406, 408) and destined for another node (see Fig. 4). The packet travels from the root complex (412) to shared I/O switch (420). At shared I/O switch (420) the packet is modified by the insertion of an OS Domain Header (e.g. 1100) to create a PCI Express+ packet. The PCI Express+ packet is forwarded to the appropriate shared I/O controller (440, 442, 444) by way of a PCI Express link (432). At the I/O controller, data from the packet is transmitted in a standard manner on the fabric to which the I/O controller interfaces (Ethernet, Fiber Channel or Other depending upon which I/O controller the PCI Express+ packet has been sent to).

Pettey does not disclose or suggest encapsulating PCI Express or PCI Express+ packets in Ethernet, Fiber Channel or Other packets. The PCI Express+ packets are modified versions of and do not encapsulate PCI Express packets.

Pettey- Compute Nodes

The Applicant notes that Pettey's shared I/O switch (e.g. 420) is not a compute node. A compute node must be capable of performing computations. The Pettey shared I/O switch (420) operates at the packet level to route packets and modify packets by adding or removing OS Domain information. Pettey does not disclose or suggest that shared I/O switch (420) performs computations that can be applied to solve any computational problem. Similarly, Pettey does not disclose or suggest that shared I/O controllers (440, 442, 444) perform computations that can be applied to solve any computational problems. Shared I/O controllers (440, 442, 444) are not themselves compute nodes.

Claim 1

Claim 1 recites "`encapsulating the local packetized interconnect packet in an inter-node communication network packet addressed to the receiving compute node`". The inter-node communication network packet containing the encapsulated local packetized interconnect packet is sent from a sending node to a receiving node. Claim 1 recites "`each of the compute nodes having an independent address space`". These features in the context of claim 1 are not disclosed by Pettey, as understood.

Claim 1 recites "`each of the compute nodes having an independent address space`". In Pettey, as understood, all PCI Express (and PCI Express+) packets remain within a single address space (corresponding to the load/store fabric).

Further Pettey, as understood, does not disclose "`encapsulating the local packetized interconnect packet in an inter-node communication network packet addressed to the receiving compute node`" as claimed in claim 1 or the overall transfer of encapsulated local packetized interconnect packets between nodes having independent address spaces by way of an inter-node communication network as recited in claim 1. Encapsulating a packet of one protocol, A, within a packet of another protocol, B, means packaging the protocol A packet (including header and data) as payload data in a protocol B packet.

Pettey, as understood, does not disclose or suggest encapsulating a PCI Express + packet inside any inter-node communication network packet. Creating a PCI Express + packet by inserting OS Domain information into the packet is not 'encapsulating' the PCI Express packet. Neither of the architectures described by Pettey provides "`encapsulating the local packetized interconnect packet in an inter-node communication network packet addressed to the receiving compute node`" as claimed in claim 1.

Pettey uses the word 'encapsulated' in the following context

> "It is also envisioned that the addition of a header within a load/store fabric, as described above, could be encapsulated within another load/store fabric yet to be developed, or could be encapsulated tunneled or embedded within a channel based fabric such as Advanced Switching or All Ethernet. Regardless of the fabric used downstream from the OS Domain (or root complex), the inventors consider any utilization of the method of associating a shared I/O endpoint with an OS Domain to be within the context of their invention, as long as the shared I/O endpoint is considered to be within the load/store fabric of the OS Domain" (see col. 23, ln. 35-45)

This section is contradictory and difficult to understand. It requires the shared I/O endpoint to be within the load/store fabric of the OS Domain but suggests that some other fabric could be downstream from the root complex. A careful reading of this section of Pettey indicates that it is 'the addition of a header' that could be encapsulated. Pettey is probably contemplating adding an OS Header to one of: a load/store fabric yet to be developed, a channel based fabric such as Advanced Switching, or All Ethernet. Pettey contemplates using the mechanisms of encapsulating, tunneling or embedding the OS header in one of these protocols. The words "encapsulation" and "tunneling" are apparently being used in a non-standard way. In any event, any 'encapsulation' occurs in the segment upstream from an I/O controller, within the address space of one server. This section of Pettey does not describe an inter-node communication network.

The Office Action cites Pettey at col.18, ln. 64 - col.19, ln. 11 as an example of encapsulation of a PCI Express packet. The Applicant respectfully submits that this is also incorrect. This section describes converting a PCI Express packet into a PCI Express+ packet by inserting an OS header. A PCI Express packet is not encapsulated in any other packet.

Further, PCI Express links (432) do not constitute an 'inter-node communication network'. These links are all within the load/store fabric of the originating server (see col. 3, ln. 54-56; col. 12, ln. 13-14).

For at least the reasons above, the Applicant submits that Pettey fails to anticipate claim 1.

As all of claims 2 to 38 depend from claim 1, all of claims 2 to 38 are also submitted to patentably distinguish Pettey. These dependent claims are submitted to further distinguish Pettey at least for the reasons below.

Claim 2

Pettey, as understood, fails to disclose "associating a first range of addresses in an address space of a sending one of the compute nodes with the network interface of the sending compute node and, at the network interface of the sending compute node associating the first range of addresses with the receiving compute node" as recited in claim 2. This association can permit the network interface to automatically select a destination address on the inter-node communication network to direct the encapsulated packet to the receiving node without the involvement of the sending CPU. Pettey has no equivalent function.

The cited portions of Pettey describe how: a shared Ethernet controller associates an OS Domain of a received PCI Express+ packet with operating system domain resources (col. 16, ln. 7-31); how a shared I/O switch adds an OS Domain to a PCI Express packet to create a PCI Express+ packet (discussion of Fig. 15 at col. 18, ln. 64 to col. 19, ln. 12); and how a

shared I/O switch processes PCI Express+ packets received from endpoints (discussion of Fig. 16 at col. 19, ln. 13 to 33).

The OS Domain is not 'a range of addresses in an address space of a sending compute node' and is not even an address in an address space of a sending compute node. The OS Domain is not included in PCI Express packets being sent within a server. Further, Pettey does not disclose associating the OS Domain at a network interface of a sending compute node with a specific receiving compute node.

Therefore, claim 2 is submitted to be patentable over Pettey.

Claim 3

The rejection of claim 3 states that the material in col. 18, ln. 64 to col. 19, ln. 36 of Pettey discloses the address translation recited in claim 3. The Applicant respectfully submits that this is incorrect. There is no mention of address translation in the cited material. The cited portion of Pettey discusses the conversion of a PCI Express packet into a PCI Express+ packet by adding an OS Domain. Inserting OS Domain information is not address translation. Pettey specifically indicates that the PCI Express packets and PCI Express+ packets always remain within the same load/store domain (see e.g. col. 12, ln. 13). This indicates that the address initially specified in a PCI Express packet remains valid at all times through conversion of the packet to a PCI Express+ packet.

Therefore, claim 3 is submitted to be patentable over Pettey.

Claims 4, 5 and 10

The rejection of claim 4 states that material in col. 16, ln. 7 to 31 of Pettey discloses the address translation recited in claim 4. The Applicant respectfully submits that this is incorrect.

Claim 4 recites "editing the local packetized interconnect packet by changing an address to which the packet is addressed from an

address in the first range of addresses to a corresponding address in an address space of the receiving compute node"

The cited portion of Pettey describes the assignment of addresses to OS Domains on initialization of PCI Configuration logic (1354) of a shared I/O controller (1300). Each OS domain can then communicate with the I/O controller using the assigned address 'to view it as a controller having resources that are dedicated to its OS Domain' (col. 16, ln. 26-7). No address translation is necessary. Once addresses are assigned at initialization, load/store operations can proceed to access memory and devices using the assigned addresses. The address assignments remain valid until the system is shut down.

Further, the initialization-time address assignment contemplated by Pettey is being done within the context of a single address space. In contrast, the address translation recited in claim 4 is from an address in the first range of addresses [which are in an address space of the sending compute node] to a corresponding address in an address space of the receiving compute node. Pettey does not disclose of suggest such address translations.

Therefore, claim 4 is submitted to patentably distinguish Pettey. Claims 5 and 10 depend from claim 4 and are submitted to be patentable over Pettey for at least this reason.

Claim 6 to 9

Claim 6 recites "allocating a region of the memory of the receiving compute node to receive data from the sending compute node, memory locations in the allocated region being addressable by addresses in a second range of addresses corresponding to the first range of addresses, and communicating the second range of addresses from the receiving compute node to the sending compute node prior to tunneling the data from the sending compute node to the receiving compute node". The rejection of claim 6 is based on col. 16, ln. 7 to 31 of Pettey and col. 18, ln 64 col. 19, ln. 36 of Pettey.

As discussed above, col. 16, ln. 7 to 31 describes the assignment of addresses for an I/O controller at system startup within a single address space and col. 18, ln. 64 to col. 19, ln. 36 describes packet transmission from the shared I/O switch to the I/O endpoint and from the I/O endpoint to the shared I/O switch. Neither of these sections describes assignment of a second range of addresses in an address space of a receiving compute node with a first ranges of addresses in a address space of a sending compute node.

Further, the Pettey shared I/O switch can not be considered to be a compute node. This switch directs received packets to specific output ports and modifies the packets in flight by adding or removing OS Domain Headers. Pettey, as understood, fails to disclose or suggest that the shared I/O switch performs any data processing.

Therefore, claim 6 is submitted to patentably distinguish Pettey. Claims 7 to 9 depend from claim 6 and are submitted to be patentable over Pettey for at least this reason.

Further, in relation to claim 9, Pettey fails to disclose allocating the claimed ranges of addresses and does not disclose first and second ranges of addresses that are the same size, as claimed.

Therefore, all of claims 6 to 9 are submitted to be patentable over Pettey.

Claims 11 and 19

The rejections of claims 11 and 19 are based on the material in Pettey col. 16, ln. 7 to 31 and col. 18, ln. 64 to col. 19, ln. 36. The Office Action indicates that these sections disclose read request packets and write request packets. The Applicant has carefully reviewed the cited sections of Pettey and can find no reference to read request packets or write request packets. The Applicant submits that Pettey fails to disclose the features of claim 11 or 19.

Therefore, claims 11 and 19 are submitted to be patentable over Pettey.

Claims 20 - 22

Claims 20-22 depend from claim 2 and therefore distinguish Pettey.

Further, Claim 20 recites "at the network interface of the sending compute node, associating the first range of addresses with a plurality of receiving compute nodes". The Office Action alleges that this feature is provided by col. 16, ln. 7 to 31. The Applicant respectfully submits that this is incorrect.

Pettey, as understood, does not disclose associating a range of addresses with a plurality of compute nodes. The cited portion of Pettey describes PCI configuration logic which initializes addresses to be used by different OS Domains. This section describes the option of providing 'a hard coded machine address ... for each of the 1-N OS Domains ...'. A hard-coded machine address is not a range of addresses. Further, if one attempts to equate the Pettey I/O controller with the network interface of a sending compute node then the OS Domains are not 'receiving compute nodes'.

Therefore, all of claims 20 to 22 are submitted to be patentable over Pettey.

Claim 23

Claim 23 recites dispatching a packet to a plurality of receiving nodes by way of a multicast feature. Multicast involves the delivery of the same information to a group of destinations. The Office Action alleges that Pettey discloses the feature of claim 23 in col. 6, ln. 5 to 24. The Applicant respectfully submits that this is incorrect. The cited portion of Pettey discloses how two nodes can share an Ethernet I/O controller via the shared I/O switch. The quoted material describes the two nodes each sending packets to the Ethernet I/O controller. There is no mention of multicast and there is no mention of one node sending packets to multiple receiving nodes as recited in claim 23.

Therefore, claim 23 is submitted to be patentable over Pettey.

Claim 24

Claim 24 recites "at the sending compute node, encapsulating each of a plurality of copies of the local packetized interconnect packet in an inter-node communication network packet addressed to a different one of the plurality of receiving compute nodes and dispatching each of the inter-node communication network packets to the corresponding receiving compute node by way of the inter-node communication network". The Office Action alleges that this feature is disclosed at col. 16, ln. 7 to 31. The Applicant has carefully reviewed this section of Pettey and can find no mention of the feature of claim 24 (or any mention of a plurality of copies of a packet generally) in this or any other section of Pettey.

Therefore, claim 24 is submitted to be patentable over Pettey.

Claims 25 to 26

Claim 25 recites "encapsulating the plurality of local packetized interconnect packets in the same inter-node communication network packet". The Office Action alleges that this feature is disclosed at col. 16, ln. 7 to 31. The Applicant has carefully reviewed this section of Pettey and can find no mention of the feature of claim 25 (or any mention of encapsulating a plurality of packets in another packet) in this or any other section of Pettey.

Therefore, claim 25 is submitted to be patentable over Pettey as is claim 26 which depends from claim 25.

Claim 29

The rejection of claim 29 alleges that Pettey discloses an atomic read-modify-write packet at col. 14, ln. 32. The Applicant respectfully submits that this is incorrect. There is no mention of atomic read-modify-write packets or any remotely equivalent concept at or near the cited line of Pettey.

Therefore, claim 29 is submitted to be patentable over Pettey.

Claim 32

Claim 32 recites "at the receiving compute node, altering a correspondence of the first range of addresses to addresses in the address space of the receiving compute node by changing the second range of addresses to a third range of addresses". The Office Action alleges that this feature is disclosed by Pettey at col. 16, ln. 7 - 31. The Applicant respectfully submits that this is incorrect. As discussed above, the cited portion of in Pettey describes assignment of addresses for an I/O controller on initialization. The cited portion of Pettey does not contemplate altering a previously-established correspondence of addresses, as claimed.

Therefore, claim 32 is submitted to be patentable over Pettey.

Claim 33

Claim 33 recites "converting the read response packet into a write request packet". The Office Action alleges that this feature is disclosed by Pettey at col. 18, ln. 55 to col. 19, ln. 36. The Applicant respectfully submits that this is incorrect. The cited portion of Pettey describes associating and disassociating OS Domains with packets. There is no mention of converting a read response packet into a write request packet in this or any other portion of Pettey, as understood.

Therefore, claim 33 is submitted to be patentable over Pettey.

Claim 35 and 36

Claim 35 recites "allocating a memory ID to a range of addresses in an address space of the receiving compute node" and "associating a corresponding first range of addresses in an address space of the sending compute node with the memory ID and the receiving compute node" and "at the sending compute node, determining an offset from

an address associated with the local packetized interconnect packet and the first range of addresses and passing the memory ID and the offset in the inter-node communication network packet to the receiving compute node". The Office Action alleges that Pettey discloses this entire combination of features at col. 18, ln. 55 to col. 19, ln. 36. The Applicant respectfully submits that this is incorrect. The cited portion of Pettey describes sending a PCI Express+ packet from the shared I/O switch to the I/O endpoint and sending a PCI Express+ packet from the I/O endpoint to the shared I/O switch. There is no mention of memory IDs, determining offsets from addresses of local packetized interconnect packets, or sending memory ID and offset by way of an inter-node communication network in the cited section of Pettey or, to the applicant's understanding, anywhere else in Pettey.

Claims 35 and 36 are therefore submitted to be patentable over Pettey.

Claim 39

Claim 39 recites "in response to determining that the packets are addressed to addresses in the first range of addresses, encapsulating the packets in inter-node communication network packets addressed to the receiving compute node" and "dispatching the inter-node communication network packets to the receiving compute node by way of the inter-node communication network". As described above in respect of claim 1, Pettey fails to disclose or suggest these features.

Therefore, claim 39 and claims 40-42 which depend from claim 39 are submitted to be patentable over Pettey.

Claim 41

Claim 41 recites "performing an address translation on the local packetized interconnect packet after receiving the packet at the network interface of the sending compute node and prior to placing the extracted packet onto the local packetized

interconnect of the receiving compute node". Pettey fails to disclose an address translation as recited in claim 41 (see discussion above regarding claims 3 and 4).

Therefore, claim 41 is submitted to be patentable over Pettey.

Compliance with 35 U.S.C. §103

The Office Action cites US patent 7,024,613 (Carnevale) in combination with Pettey in relation to claims 12 to 18. The Applicant submits that claims 12 to 18 are patentable over the cited combination.

Claims 12 to 18 depend from claim 1. Carnevale does not remedy the above-noted defects of Pettey. For at least this reason, claims 12 to 18 are not rendered obvious by the cited combination of Pettey with Carnevale.

Claims 12 to 18 are further submitted to distinguish Carnevale for the reasons set out below.

Carnevale

Carnevale relates to InfiniBand networks. In such networks, links interconnect components such as host bus adapters (HBAs), switches, routers, and target channel adapters (TCAs). HBAs nominally act as I/O controllers for a host CPU. TCAs act as InfiniBand network interfaces for a resource that is being used by a host CPU. A host CPU typically sends an I/O request through an HBA, across the InfiniBand network, and through a TCA to the resource. The HBA acts on behalf of the host CPU as an InfiniBand requester and the TCA acts on behalf of the resource as an InfiniBand responder. Communication between HBAs and TCAs utilize queue pairs (QPs) (i.e. pairs of queues). In a QP, one queue holds packets that are queued awaiting transmission across the InfiniBand network. The other queue holds packets that have just been received from the InfiniBand network and have not yet been forwarded on to the host CPU or resource.

Carnevale describes particular ways to implement queue pairs. A series of pointers are associated with a transmit queue. The pointers represent progressive states in the life cycle of a request. Pointers advance as requests and packets are dealt with.

Claim 12

Claim 12 recites "generating a write confirmation packet at the network interface of the sending compute node and dispatching the write confirmation packet on the local packetized interconnect of the sending compute node". The Office Action indicates that this feature is disclosed in Carnevale at col. 4, ln. 1 - 24. The Applicant respectfully submits that this is incorrect.

The quoted material in Carnevale describes acknowledge packets being sent from an IB responder back to an IB requester (see col. 3, ln. 67 to col. 4, ln. 1). The IB responder is at a receiving end of the IB link. Carnevale, as understood, fails to disclose the feature of claim 12.

For this additional reason, Claim 12 is submitted to patentably distinguish the cited combination of Pettey and Carnevale.

Claims 14 to 16

Claim 14 recites "changing the write confirmation packet by altering the contained address of the memory location in the address space of the receiving compute node to a corresponding address in the first range of addresses". Carnevale, as understood, fails to disclose or suggest this feature. As noted above, Pettey also fails to disclose this feature.

For this additional reason, Claim 14, and claims 15-16 which depend from claim 14 are submitted to patentably distinguish the cited combination of Pettey and Carnevale.

Conclusion

The applicant submits that pending claims 1 to 46, as amended, are in condition for allowance. Reconsideration and allowance of this application are respectfully requested.

Respectfully submitted,

By: /RichardAJohnson/
Richard A. Johnson
Registration No. 56,080
tel: 604.669.3432 ext. 9046
fax: 604.681.4081
e-mail: GNMDocket@patentable.com